CONF-IRM 2015 Proceedings

International Conference on Information Resources Management (CONF-IRM)

5-2015

Profiling Behavior of Intruders on Enterprise Honeynet: Deployment and Analysis

Ling Xue
City of Keene, lxue@ksc.keene.edu

Wei Lu
Keene State College, wlu@keene.edu

R61. Profiling Behavior of Intruders on Enterprise Honeynet: Deployment and Analysis

Ling Xue
City of Keene
lxue@ksc.keene.edu

Wei Lu
Keene State College
wlu@keene.edu

Abstract

Network and information security continues to be one of the largest areas that require greater attention and improvement over the current state of infrastructure within enterprise information systems. Intruders to enterprise networks are no longer just hacking for fun or to show off their programming skills; rather they are now doing it for profit-making motives. As a result, developing profiles for the behavior of intruders, trespassing upon business information systems within an enterprise networking environment, has become a primary focus of cyber-security research recently. In the proposed on-going project, we deploy a novel honeynet system using advanced virtualization technologies, in order to collect the forensic evidence of an attack, by allowing attackers to interact with compromised computers in a real enterprise network. We then analyze the behavior of intruders in order to investigate and compare their hidden linkages as compared with enterprise networks, and the attacker(s)' potential group structures, including attributes such as geographic distribution and service communities, thus providing strategies for enterprise-network administrators to stay protected against malicious attacks from external intruders. Preliminary results on the proposed research is very promising, showing intruders' behaviors over one month were distributed across over 60 different countries, and our work demonstrated that the most popular service intruders like use to interact with is the very HTTP Web itself.

Keywords

Honeynet, Network Security, Enterprise Information System, Virtualization

1. Introduction

Honeynets provide an information-gathering approach to security; they are used to gather information about threats in the network. A honeynet is an interactive type of honeypot which provides real systems and application for intruders to attack and thus captures real information on a real attack. A honeynet is based on the idea of deception (Warkentin 2006), in that the hackers are tricked into thinking that they are interacting with the real production systems; hence they can do all the damage thinking that they are not being watched.

That collected information is then used for improving the detection, defense, and mitigation techniques employed against those monitored types of attacks. In this context honeynets are not only important for attracting attackers so as to capture great amounts information on the activities of the black-hat community, but a honeynet project in the enterprise system can help the information system manger enforcing better security policies on the basis of vulnerabilities found by honeynet, thus developing different security aspects involved in setting up and maintaining a honeynet on a digitalized enterprise networking system (Romney et al. 2005).

In the enterprise information system, honeypots are typically used to add another layer of security to the network, as more-conventional firewalls and network-intrusion detection systems (NIDS) have some limitations including, for example: (1) The network firewall placed at the network's external interface to the Internet cannot protect hosts within the corporation from threats that originate within the corporate network (intranet attacks), and (2) some attacks can manage to bypass the firewall, while the NIDSs usually suffer from large numbers of false positives and false negatives (Russell 2000). The increasing use of encryption also reduces the amount of useful information that can be collected within a NIDS; plus, a NIDS may fail to detect new attacks having signatures unknown in its forensic database (Provos and Holz 2007).

Therefore deploying a honeynet to capture real attacker actions can help acquire the information missed by the firewall and NIDS. Furthermore, this information can be used to refine NIDS rules, in order to reduce the number of false positives, thus helping to deploy stronger security measures for protecting enterprise information system from internal threats (Honeynet 2004).

Toward the assistance of government regulations regarding information security, the collected information by a honeynet on an enterprise network is very useful for fighting and identifying criminals such as those precipitating distributed-denial-of-service (DDoS) attacks, spamming, phishing, keylogging, click fraud, as well as identify theft and information ex-filtration. Precedent has shown that such recorded evidence regarding what intruders are doing, and what their ultimate goals are, can be brought up in court toward convicting intruders of the enterprise network. Moreover predicting intruders' actions on the basis of data collected by the honeynet can mitigate the risks of being attacked.

One of the big challenges in deploying honeynets on an enterprise system is how costly and difficult-to-maintain they can be, when there are many different real systems and applications involved. In this research-in-progress, we propose a novel virtual-honeynet system based on the advanced hardware-virtualization (VM) technologies in which the fingerprinting of the honeypot systems cannot be spied out by intruders, thus protecting the whole honeynet system from direct attack by those very intruders it seeks to study.

2. Overview of Honeypot and Honeynet

Honeypots can be classified into two major groups, namely low-interaction honeypots, and high-interaction honeypots. The difference between these two classes is the extent to which the attacker is allowed to interact with the system. As the words suggest, a low-interaction honeypot keeps suspected-attacker contact with the physical system to a minimum, by way of emulated systems and applications for the attacker to target and typically the use of mere scripts to respond to the intruders activities. The major advantage of low-interaction honeypots is that they are simple to make and easier to deploy, and the information gathered is mostly statistical data on port-scan and worm attacks, plus new and ongoing attack patterns. A high-interaction honeypot on the other hand, is the one that uses real systems and applications to interact with the attackers. A network which employs honeypots of this type is what we call a honeynet. These are simply systems running real operating systems such as Windows Server, Linux that are placed on a real network as honeypots and then exposed (but not via the usual contrived weaknesses) to the hackers to be attacked. A great deal of valuable information can be collected from the attackers' high degree of interaction with such a honeynet. For the accuracy and integrity of the data collected, the honeynet systems should not be intentionally compromised (or configured) in any way that will advertise them to or make them more vulnerable to attackers (Currant 2005). Limitations with high-interaction honeypots are high maintenance and close monitoring, as well as the need to analyze the attacker's behaviors and attempt to find the motivation of their attacks, which can be quite time consuming.

A physical honeynet is one in which the honeypots applications are running on separate physical machines, rather than being run as several virtual hosts in a single physical machine; in contrast to a physical honeynet, a virtual honeynet is a technology that virtually implements many different operating systems in one hardware computer, and hence rather than having a honeynet containing different physically separate honeypots, all the honeypots will be virtually housed in one machine and still appear to the attacker as different separate machines. The main advantage of using a virtual honeynet over a classic physical honeynet is the cost of equipment: a virtual honeynet often needs only one physical machine, rather than requiring the security researcher to buy many physical machines and their connecting equipment. It is also easy to manage since all the honeypots are configurable in-and-from one physical machine.

There are two typical types of virtual honeynets: one is self-contained and the other one is hybrid. The self-contained virtual honeynet is where the whole honeynet, which includes honeypots and the honeywall and any other data control and data capture tools, is contained in only one physical computer. It is advantageous because it is portable to carry the whole honeynet around from place to place. Furthermore using only one hardware device cuts much on the cost of buy-ing many separate computers. The main drawback of self-contained honeynet however is that since all the services are run in one machine if that machine fails or is compromised then the whole honeynet is gone, and if the attacker detects they are interacting with a self-contained virtual honeynet they can then attack other parts like the firewall, the IDS since they are all in the same system. Also there is going to be a great need of memory and CPU power in that machine, hence the cost of upgrading its CPU and memory may be too high.

We are therefore proposing a new type of hybrid-virtual honeynet, in which control and capture of data are implemented in physically separate machines and the honeypots are then run virtually on the same computer. This approach is more robust than the purely physical, self-contained honeynet model, and is very flexible as there are no limitations on the type of technologies to use for data control and data capture.

3. The Proposed Virtual Honeynet

Figure 1 illustrates the networking topology of our honeynet architecture, following which we describe the main functions for each component.

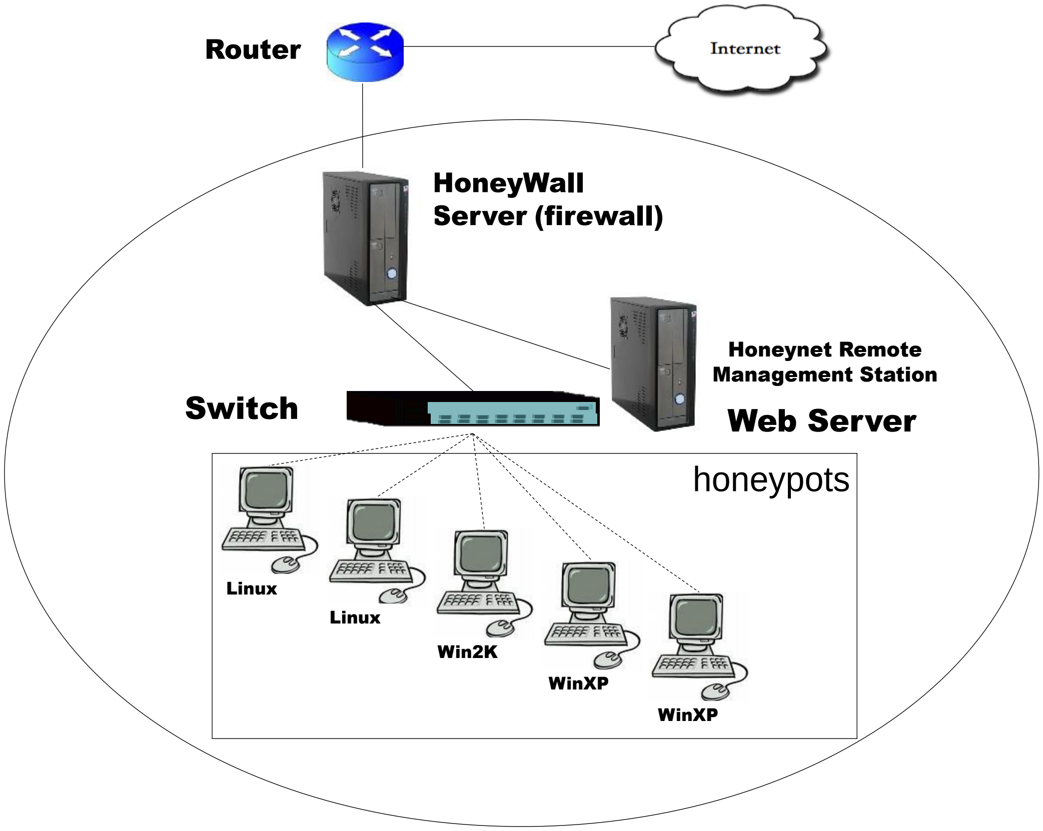

Figure 1. Hybrid Virtual Honeynet Architecture and Technology

3.1 Data Control

Data control is the most important function when implementing a honeynet, as this function is providing an environment for the attacker to carry out the attack activities, with as much freedom as possible, and without noticing that he or she is interacting with a honeynet. Data control of a honeynet also has to prevent the attacker to use the honeynet from targeting other non-honeynet machines either by mistake (if the intruder means only to target computers on the single network) or intentionally, even when the honeynet itself is compromised. In our virtual honeynet we implement different data-control layers (such as counting outbound connections, intrusion prevention gateways, or bandwidth restrictions), so that our system can avoid a single point-of-failure, especially for new types of attacks. This module also ensures that our honeynet blocks all outbound connections to the real production systems (or to the rest of the Internet, for that matter) in the event of software or hardware failures within the honeynet.

3.2 Data Capture

Data-capture functionality attempts to record as much data and details (metadata) as possible from the attackers' actions by monitoring and logging all their activities while ensuring that such monitoring is imperceptible to the intruder. All activities must be captured including activities on the network, against the honeynet hosts, as well as those activities that originate within the honeynet. One of the major challenges faced by data-capture is the use of encryption (e.g. in SSH sessions) by the black-hat community. Merely sabotaging the honeypot’s ability to receive encrypted packets by shortening the length of its public session-keys would be too conspicuous to attackers. Likewise producing weak public session-keys though reducing the quality of their hash upon the private key would still be too obvious, as well. This means that encrypted data has to be captured in the unencrypted form. This can obviously be done by simply trying to break (brute-force) the public session-keys and then decrypt the data, but to do so is exceedingly difficult because the attacker expects us to be using keys of sufficient length (e.g. 256-bit or more). Another approach is to capture the data post-decryption within the honeypot’s operating system, which is indeed the technique used by the kernel-based Sebek tool used for data-capture by the honeynet project. Using different layers to capture this data can be beneficial since if one system missed some details, then it remains possible for us to capture the packets with one the others. Also, if the attacker is able to detect that he or she is being monitored, and tries to bring down the capturing tool (e.g. by disabling or bypassing Snort somewhere) then we are still able to capture those details, and others, with the remaining tools. In our honeynet we apply Sebek, IDSs, and firewall logs, to conduct data capture.

3.2.1 Using the Sebek data-capture tool

Sebek is a data capturing tool that resides in kernel space and is able to capture some data on user activities as they access the system. It is able to copy most or the entire intruder’s activities from the honeypot(s) and transfer it over the honeynet network to the honeywall anonymously without the intruder detecting this. The Sebek tool does keystroke logging for encrypted sessions, and it is able to retrieve passwords that intruders use for remote logins, retrieve input of keys to decrypt the encrypted data, and to recover files copied over Session-Control Protocol (SCP) (Dornseif, 2004). When encryption is not used, it is easier to get the intruder’s keystroke actions and user output details using a tool like Ethreal which can sniff packets in the wire and then do some stream reassembly to determine the TCP sessions of the intruder. Using Sebek in the operating system’s kernel space enables administrators to capture any intruder’s activities regardless of what binaries the intruders are using for decryption. Also because user space is separate from kernel space, the Sebek instance can be hidden from all users, including those with root privileges.

3.2.2 An intrusion-detection system

In our honeynet we use Snort, an open-source Intrusion Detection System (IDS) to monitor and log all traffic in the honeynet. Snort sits between the honeynet and the external network, examining and reporting (including generating alerts) all traffic according to a pre-defined configuration of rules. In our honeynet project, Snort is set to a full-logging mode so that it captures all the IP packets, plus TCPdump is employed to capture all the network traffic in binary format. So whenever the snort instance fails, the data missed is still fully recorded by TCPdump. Another type of IDS we deploy is Bro, the functions of which are mostly anomaly-detection-based, and can be used to complement Snort though capturing additional details, as well as for correlation of results.

3.2.3 Firewall logging

The firewall monitors all traffic entering and leaving the honeynet, so logs of this traffic can provide very useful data for analysis (e.g. an attempted backdoor access to the honeynet). The firewall is able to log the failed or refused connections, which means that some details missed by other tools can be apprehended by these logs.

4. Deployment and Analysis

There are three options for where we can place our honeynet in the target network, relative to other networking devices, which are either externally facing the Internet, internally behind the firewall, or in the demilitarized-zone (DMZ) subnet (Grimes 2005). Each of these locations has their own advantages and disadvantages, and the choice depends on factors including amount of network resources the enterprise has, and the objectives of deploying a honeynet.

For example, a research-based institution that wants to capture as much hacker details as possible will likely put their honeynet externally, whereas an organization that wants to have up-to-date details of the possible exploits to their production systems will likely place their honeynet internally. The expertise of the administrators can also be a factor in this decision, as some deployments are more complex than others.

In this ongoing research project we place our honeynet on the enterprise DMZ, in the same sub-net alongside some legitimate public access DMZ servers (e.g. web and email servers), thus being able to capture early details of attacks to devices on the DMZ. Deploying honeynet in the DMZ subnet enforces more robust data-control measures compared to the other two options, as doing so works by way of placing an additional router between the honeynet and the DMZ firewall for controlling the outgoing connections from the honeypots, and therefore the production networks protected behind the DMZ are at less risk of being attacked.

We run our honeynet on an enterprise network for one month during which we received more than 200,000 packets. The total number of unique source IP addresses reaching our honeynet was originated from 62 countries. Most of the IP addresses belong to United States and China. Port 80 was the most common destination port according to our experimental analysis of the honeynet traffic collected over one month, i.e. the most popular service hackers like to interact with is port 80, as that is the port assigned to HTTP traffic, the basis of the World Wide Web.

5. Conclusions

In this proposed continuation of our research-in-progress, we aim to investigate existing work in the area of forensics upon individual hackers and hackers' organizational structures, in order to develop a novel hybrid-virtual honeynet technique with which to collect the forensic evidence of attacks by allowing attackers to interact with compromised computers in a real enterprise network. To the best of our knowledge, we believe the proposed work is the first attempt to address a key gap in the discipline of Network Security, by mapping the linkages between the characteristics of attacking traffic and their possible structure hidden behind, thus allowing the average enterprise systems-administrator to keep protected against malicious attacks from computer intruders.

References

Warkentin, M. (2006). *Enterprise Information Systems Assurance and System Security: Managerial and Technical Issues.* Hershey, PA, USA: Idea Group Publishing, 2006.

Romney, G. W. et al. (2005). “IT Security Education is Enhanced by Analyzing Honeynet Data,” In Proceedings of the 6th International Conference on Information Technology Based Higher Education and Training (ITHET 2005).pp. F3D-10 - F3D-14.

Provos, N. and Holz, T. (2007). *Virtual Honeypots from Botnet Detection to Intrusion Detection*, Addison Wesley, Boston, USA, July 2007.

The Honeynet Project (2004), *Know Your Enemy Learning About Security Threats*, Addison Wesley, Boston, USA, July 2004.

Curran, K. et al. (2005) “Monitoring hacker activity with a Honeynet.” International Journal on Network Management, January 2005, pp. 123-134.

Dornseif, M. et al. (2004) “NoSEBrEaK- Attacking Honeynets” In Proceedings of the Fifth Annual IEEE SMC Information Assurance Workshop, June 2004, pp. 123-129.

Grimes, R. A. *Honeypots for Windows*, Kinetic Publishing Services, 2005

Russell, R. (2000). *Hack Proofing Your E-Commerce Site: The Only Way to Stop a Hacker Is to Think Like One*. Rockland, MA, USA: Syngress Publishing, 2000.